METHOD OF PRE-PROCESSING DATA ON A HOST MACHINE FOR A HOST-BASED PRINT SYSTEM

BACKGROUND

[0001] Prior solutions typically use original print data to generate single bit print data on a host machine, e.g. computer. Reducing the print data to a single bit stream enables faster transfer of the data between the host machine and the printing device. While this allows for higher print speeds, there is significant loss of image detail.

[0002] To compensate for the loss of image detail, extensive processing is done on the printing device to reconstruct the original image detail using the single bit data stream. This is typically done with extensive hardware circuits and or high-powered processors. These hardware resources add significant cost to the printing device.

[0003] Figure 1 illustrates an overview process flowchart according the prior art described above. On the host machine, an application, e.g. Microsoft PowerPoint, generates page data at a defined pixel rate, e.g. 8 bits per plane per pixel. The image processor renders page data to 1 bit per plane per pixel. While this will allow for faster data transfer to the printer, much of the image detail is lost. The image processed data may be further compressed, e.g. JBIG or JPEG, to facilitate data transfer. The host I/O manages the data to and from the printer.

[0004] The printer I/O manages data to and from the host machine. If the received data has been compressed, it will be uncompressed according to the appropriate decompression algorithm. The imaging processing hardware attempts to recreate the details of the original image data using 1 bit per plane per pixel data. A pulse code generator generates pulse codes to control the laser using the modified image data from the image processing hardware. The print engine mechanism uses the pulse codes to print hardcopy image.

[0005] Figure 2 illustrates a functional block diagram of the host machine according to the prior art described in Figure 1. The application interacts with the operating system (OS) Graphics Interface. A spooler stores data, received via the OS Graphics Interface, in object form. The print processor reads graphics objects stored by the spooler and generates a raster bitmap. The bitmap contains 1-bit per pixel values.

SUMMARY

[0006] In the present invention, the processing required to generate high quality print output is shifted from the printing device to the host machine. This allows for minimal hardware resources on the printing device.

[0007] In one embodiment, a printing system comprises a host machine and a printing device. The host machine includes an application, image processor, optional controller, and an I/O device. The application generates page data. An image processor receives the page data and renders interim data that contains a multi-bit value per plane per pixel. The interim data may be optionally compressed. I/O device transfers the data to the printing device. The printing device includes an I/O device, an optional controller, a pulse code generator, and a print engine mechanism. The I/O device receives data from the host machine. The controller decompresses the data, if compressed. The pulse code generator, receives the interim data, for each multi-bit value, then generates a pulse code corresponding to the multi-bit value. The print engine receives the pulse codes and transforms the pulse code to a hardcopy image.

BRIEF DESCRIPTION OF THE DRAWINGS

[0008] Figure 1 illustrates an overview process flowchart according to the prior art.

[0009] Figure 2 illustrates a functional block diagram of the host machine according to the prior art described in Figure 1.

[0010] Figure 3 illustrates an overview process flowchart according to the present invention.

[0011] Figure 4 illustrates a functional block diagram of the host machine according to the flowchart described in Figure 3.

DETAILED DESCRIPTION

[0012] The interim data accurately represents the original print data without losing necessary detail while significantly reducing the overall size of the print data. The interim, multi-bit values are generated in a way that allows the printing device to easily convert them into high quality print output.

[0013] In operation, the software on the host machine examines a pixel, either alone or in relation to adjacent pixels. Based on the image data contained in the pixel or group of pixels, a multi-bit value is generated that can be used by the printing device to easily reproduce the necessary detail of the original pixel. The multi-bit value also contains print engine control parameters that effect print quality enhancements such as toner miser mode, toner explosion, edge roll off, etc. The multi-bit value can be generated algorithmically by using a lookup table or by some combination of these methods.

[0014] Figure 3 illustrates a process flowchart 100 according to the present invention.

[0015] Steps 110-140 are accomplished on the host machine. In step 110, an application generates page data at a selected bit rate per plane per pixel, e.g. 8 bits. In step 120, the imaging processor renders data to a multi-bit per plane per pixel interim value. This interim value contains all necessary image detail as well as print engine control parameters to handle print quality items, e.g. toner miser and resolution enhancement. In step 130, the data is optionally compressed to facilitate data transfer. Compression can be done using a lossless algorithm, e.g. JBIG, or by using a lossy algorithm, e.g. JPEG. In step 140, the interim data is transferred to and from the printing device.

[0016] Steps 150-180 are accomplished on the printing device. In step 150, data is transferred to and from the host machine. In step 160, if the data has been compressed, the data will be decompressed using the same algorithm that was used to compress the data. In step 170, the pulse codes are generated from the interim multi-bit values since it contains all necessary image detail and print engine control parameters. In step 180, the pulse codes are used to print a hardcopy image.

[0017] Figure 4 illustrates a functional block diagram of the host machine according to the invention described in Figure 3. The application 10 interacts with the operating system (OS) Graphics Interface 12. A spooler 14 stores data, received via the OS Graphics Interface 12, in object form. The print processor 16 reads graphics objects stored by the spooler 14 and generates a raster bitmap. The bitmap contains 8-bit per pixel values. Any print processor that generates an 8-bit per pixel values maybe used, e.g. that offered by Microsoft or Hewlett-Packard.

[0018] In the present invention, the print processor 16 examines the pixels in the raster bitmap, either individually or in relation to neighboring pixels. After this analysis, a corresponding multi-bit value (includes 2 and 4 bit values) that accurately represents the original pixel is generated. The multi-bit value may be generated using a look-up table or algorithmically. The multi-bit values can also encapsulate print engine control information to handle print quality items including toner explosion, toner miser, and resolution enhancement.

[0019] The pulse codes are coded data used to instruct the Pulse Width Modulator (PWM) block on what to do for a single pixel. Pulse codes can contain pulse position information (i.e. left, right, center, split) and pulse width information (i.e. make the pulse 15/32 of a pixel wide).

[0020] Toner Explosion is a problem with certain patterns of toner on certain types of media under certain ambient conditions that causes some of the toner to relocate on the page - the page ends up looking like tiny explosives were set off under the toner.

[0021] Toner Miser allows the reduction of the amount of toner used to render the image on a page. This is usually associated with a 'draft' mode and the image quality is usually less than that of a normally printed page.

[0022] In Resolution Enhancement, the apparent resolution of the output image is increased. It also reduces the 'jagged' edges caused by converting an image to a discrete pattern of 1's and 0's. Resolution enhancement is also used to reduce the stair step edges in slanted lines and also smooth out curved edges.